Issue 24 | Oct 2009

Blender learning made easy

blender art MAGAZINE

From out of the Deep

Tutorial - A quick little Whale

Tutorial - Realistic Water Environment

Making of - Kaldewei

Making of - Sea Anemone

COVERART Bonding - by Terence Gomez

blender art MAGAZINE

EDITOR
Gaurav Nawani gaurav@blenderart.org

MANAGING EDITOR
Sandra Gilbert sandra@blenderart.org

WEBSITE
Nam Pham nam@blenderart.org

DESIGNER
Gaurav, Sandra, Alex

PROOFERS
Brian C. Treacy
Bruce Westfall
Daniel Hand
Daniel Mate
Gaurav Nawani
Henriël Veldtmann
Joshua Leung
Joshua Scotton
Kevin Braun
Mark Warren
Noah Summers
Patrick ODonnell
Phillip
Ronan Posnic
Scott Hill
Wade Bick

WRITERS
Sandra Gilbert
SpewBoy
Christoph Aka Dracio
Arland B. Woodham III
Thomas Schlüter

COVER ART
Bonding - by Terence Gomez

CONTENTS

Sandra Gilbert
Managing Editor

Dancing beams of light bouncing off of underwater life and hypnotic light patterns shimmering on the sea floor.

"From out of the Deep..." conjures up images of all types of sea creatures, from the simple to the beautifully exotic. Dancing beams of light bouncing off of underwater life and hypnotic light patterns shimmering on the sea floor.

But the phrase also brings to mind the beautiful and unique creations that rise from the deep pools of our own creativity. And this community has an overabundance of creativity and talent.

Over the last four years, (and yes, it really has been four years) we have seen some of the best our community has to offer. From the insightful tutorials to the explanations and discussions of the numerous and varied personal and professional projects. And no issue of Blenderart Magazine would be complete without the amazing images that make their way into our galleries.

It is thanks to all these amazing artists and their willingness to share not only their knowledge, but their artwork, that our magazine has become so popular among Blender users worldwide.

So here is a big thank-you and an even bigger hug for all you have contributed and given to Blenderart Magazine and to the community at large.

And just a final note before you all dive into our latest issue. Users young and old, new and experienced learn from and enjoy the unique blend of articles, tutorials and images we publish. So let's keep the momentum going ever forward with continued submissions from all of the talented members of our community. We are waiting to hear from you. :)

sandra@blenderart.org

Blender, itself runs on just about any system you might have and you don't have have to have the most amazing computer to take advantage of Blender's capabilities.

Introduction

I recently received an email from Shaskank Sondi asking if I could do an article on what it takes to create photo-realistic renders in Blender. Specifically he wanted information on:

- Various tools used
- Minimum system requirements
- Techniques for rendering

Now I am not overly known for producing photo-realistic renders, I have a somewhat unique style of my own, that falls somewhere between toony and whatever I am in the mood for that day. But with just a little searching I came up with a number of useful resources for all you photo realistic artists out there, as well as some useful tips to keep in mind.

Let's start with system requirements. Everyone has their favorite operating system, and since the differences in final output are so minimal, honestly it boils down to what are you most comfortable with.

As for your actual hardware, that one is a bit trickier. But general guidelines would be to have the biggest processor and as much RAM as you can afford.

That being said, Blender, itself runs on just about any system you might have and you don't have to have the most amazing computer to take advantage of Blender's capabilities. Here is a list of minimum requirements as well specs for middle and high end:

Operating Systems

- Windows 2000, XP or Vista
- ac OS X 10.2 and later
- Linux 2.2.5 i386
- Linux 2.3.2 PPC
- FreeBSD 6.2 i386
- Irix 6.5 mips3
- Solaris 2.8 sparc

Minimal specs for Hardware

- 300 MHz CPU, 128 MB Ram
- 20 MB free hard disk Space
- 1024 x 768 px Display with 16 bit color
- 3 Button Mouse
- Open GL Graphics Card with 16 MB Ram

Good specs for Hardware

- 2 GHz dual CPU, 2 GB Ram
- 1920 x 1200 px Display with 24 bit color
- 3 Button Mouse
- Open GL Graphics Card with 128 or 256 MB Ram
- Production specs for Hardware
- 64 bits, Quad core CPU, 8 GB Ram
- HD 1920 x 1200 px Display with 24 bit color
- 3 Button Mouse + tablet
- Open GL Graphics Card with 768 MB Ram, ATI FireGL or Nvidia Quadro

External Renderers might require more power so check their websites for system requirements.

Its all in the Details

Photo realistic renders look convincing because the artist took the time to make it look that way. Before you even open Blender or worry about which render engine to use, you need to do research. Make sure you gather as much reference material as you can. Then sit down and really think about what you are trying to achieve. Planning is the most crucial step for a photo-realistic render.

Things you might want to think about and consider:

Materials

- What materials are needed?
- Can you create it with procedurals or do you need images, can you paint it in GIMP or Photoshop?
- Wear and tear, dirt.

Modelling

- Do you need to model every detail?
- How many details?
- Can you fake parts of it with UV mapping and good images?

Camera

- What angle are you considering?
- Are you mimicking a specific camera shot or type?

Lighting

- Where is the light coming from?
- What type of tight is it?
- The Science of CG is a must read when you are learning how to create photo realistic images.

The Science of CG is a comprehensive article on CG lighting and material properties with

great advice on achieving great results for both photo-realistic as well as non-photo-realistic images.

Various Tools

There is no one (or several) specific tool in Blender for creating photo-realistic renders. Modeling is done with the usual tools, extrude, scale and grab. You can use multi-res and the sculpt tool to add extra detail.

Probably the most useful tools in Blender for photo realistic work are the UV mapping tools and the relatively new Texture cloning tool. Using one or both of them will allow you to add/create realistic materials to enhance your models.

A beautiful example of the power of UV Mapping and Texture Cloning can be seen in MESCH973's Colibri image.

Which Render Engine?

There is always an ongoing debate concerning the best Render engine. I have no intention of adding to that debate.

The Blender internal Render engine has seen many improvements over the years and continues to improve with every release. In fact, I do believe there are current test builds with new goodies that you can test. Properly set up, Blender can produce some fairly nice photo realistic renders.

If you would prefer to use an external Render engine, there are quite a few to choose from. They are all powerful and capable of producing visually striking images. But of course each has its strengths and weaknesses. So choose the one that meets your needs.

The current popular engines are:

- Yafaray
- Kerkythea
- Luxrender
- Sunflow

Abhishek Gupta has written a great comparison article on how several of the most popular engines stack up for features and ease of use.

Kitchen tutorial by Karan Shah on CG Tuts.

If you are curious to see how you can actually create a photo realistic image/scene, I encourage you to check out the in depth Kitchen tutorial by Karan Shah on CG Tuts. He takes you from modeling all the way to rendering in Yafaray to produce a highly realistic kitchen scene.

Durian Updates

Over the last few weeks the Durian team has been busily getting settled into their apartments and running various tests on Blender 2.5 for usability and missing features.

Nathan has updated his Simple bi-ped rig to test that the rigging features were present and working. Ben has been playing with various methods of producing fire. This includes the use of volumetrics and simulation for the bulk of the fire where appropriate. Soenke has been creating faces and testing for bugs. Angela has modeled a very nice character to test that the 2.5 modeling, rigging and animation functions are present and working.

And of course all this testing has been keeping Brecht and Campbell very busy indeed. Not only putting features back, but tracking down bugs that the others are reporting.

Looks like things are rolling merrily along. For current updates on Project Durian you can follow the news on their project site and even keep up with them on Twitter.

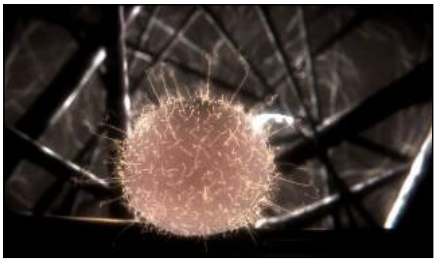
Don't forget, you can still pre-order your DVD copy of the Durian Project.

Blender Conference 2009

The eighth annual Blender Conference took place October 23-25 at "De Balie" in Amsterdam and was a fun filled 3 day adventure in Blender education, information and good times. The highlight of every Blender conference is of course the Suzanne Awards. This year's nominations were excellent and well worth checking out if you have not already done so. Congratulations to this year's winners. Go to www.blender.org for further information. Bart has set up a BlenderNation Flickr stream and is inviting all conference attendees to share their photos.

Best Character Animation

Dragosaurio - by Claudio Anduar



Best Designed Short Film

Evolution - by Alex Glawion

Best Short Film

Memory - by Ryusuke Furuya, Junichi Yamamoto

New book: 3D for iPhone Apps with Blender and SIO2

Tony Mullen announced his upcoming book on iPhone game development with Blender and the SIO2 game engine.

Tony Mullen wrote:

Hi Everybody,

I'm very pleased to announce that my latest book is available for pre-order at Amazon! The book is titled "3D for iPhone Apps with Blender and SIO2: Your Guide to Creating 3D Games and More with Open-Source Software" and you can pre-order it here.

As you can see from the title, the book deals with using Blender and the powerful SIO2 Game Engine to create 3D games and apps for the iPhone and iPod Touch. I can tell you first hand that it's a blast! The book does not assume any specific background knowledge, so in that sense it's intended for "beginners". There's even an appendix giving a quick and dirty basic introduction to Blender itself.

However, you should be aware that the material is challenging and the pace is pretty brisk, so any background you do have in Blender or graphics programming will be very helpful for you. It does assume some basic understanding of programming, so if you're new to programming in general, then you should be prepared to turn to supplemental resources to get you through any difficult patches.

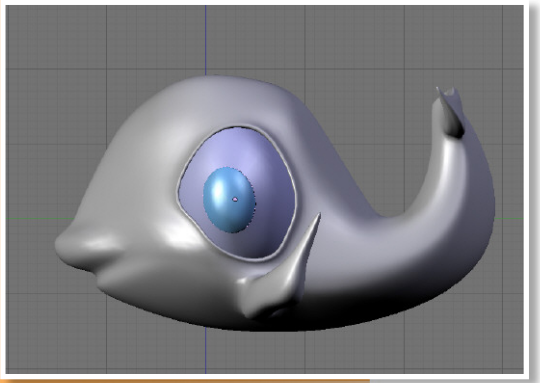
Blender 2.50 material library

Many users have wanted a material library to ship with Blender and it looks like Blender 2.50 will contain one. A call for materials has been posted on the forums.

LetterRip writes:

A material library will be added to Blender for 2.5, we would like to invite the community to participate in this by donating material settings. In order to facilitate this, please provide a link to a blend file with a single material in it.

Link - BlenderArtists thread



A Quick Little Whale

By Sandra Gilbert

Introduction

I thought it would be fun to create a quick little whale. Our whale isn't going to be overly realistic, but will give us some fun modeling practice. We are aiming to create a fairly clean mesh without a lot of unneeded vertices, and we are going to use the Grease Pencil to sketch out a very basic outline for reference.

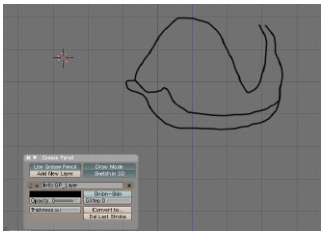
So I have this idea for a cute little whale, but my sketching skills are so rusty as to be useless, and just jumping into Blender and getting to it will likely end in frustration.

This is where the grease pencil excels. It will allow us to draw a very basic shape with just the important details, for us to use as reference.

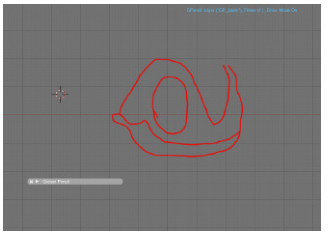
To open the Grease Pencil dialog box go to View›› Grease Pencil. There are a few options we need for this exercise. Select:

- Use Grease Pencil: This lets us see our sketch/drawing
- Draw mode: This activates the Grease pencil (it toggles it on/off)
- Sketch in 3D: This allows us to see our sketch even when rotating view

Note: Because everyone will end up with a somewhat different sketch, the following steps are some what general in nature and you may need to do some adjusting of the instructions to match your sketch.

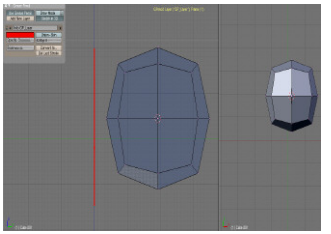


Note: I actually made life more difficult by creating my Grease Pencil sketch in front view, while of course sketching a side view of the whale. I suggest drawing in side view, then your sketch and views will match up without complications.



Step1. Okay, so I just drew a very basic outline of my little whale (1 & 2) and changed the color of my lines to red for better visibility. You'll notice, that as of yet, I only have the body drawn. We will add side fins and the tail fluke when the body is done.

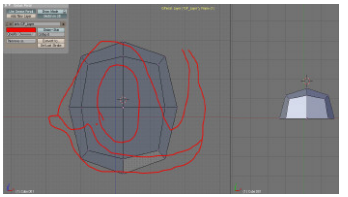
Make sure to turn off "Draw mode" when you are finished with your sketch.



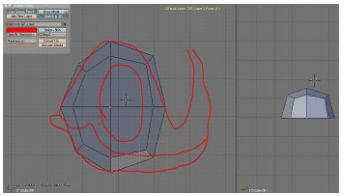
Step2. Add a cube and subdivide smooth (***W› Subdivide Smooth***).

From the front view, delete half the cube. We will add a mirror modifier later.

Step4. Scale the cube up so that it fits the height of our drawing.

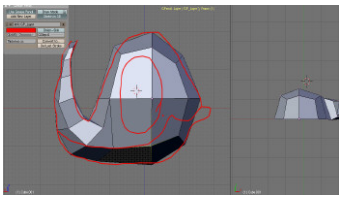


Step5. Move the cube slightly forward and start moving the outside vertices to match the whale outline.

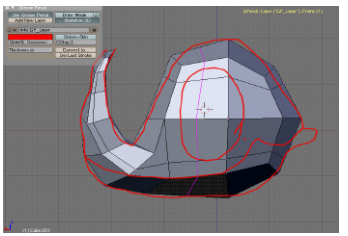


Step6. Start extruding the tail section from the back of the cube.

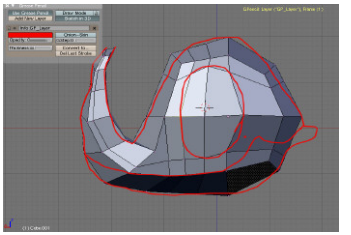
Step7. Scale and rotate each tail extrusion slightly to fit our outline.



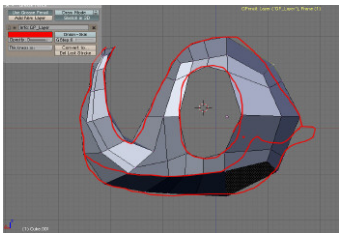
Step8. Creating the eye opening (socket). There currently aren't enough vertices to create a nice opening. So using Control + R, make a Loop cut through the eye.



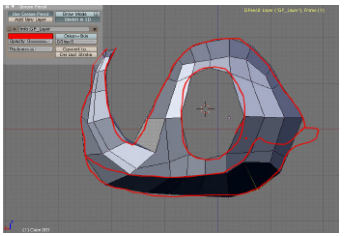
Step9. Move vertices to match the eye outline.



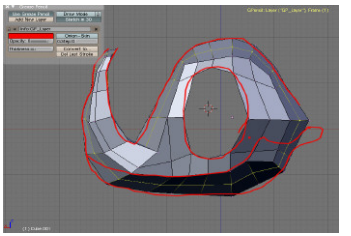
Step10. remove the vertices in the center of the eye.



Step11. Add another Loop cut in front of the eye and just start adjusting the vertices to match the image.



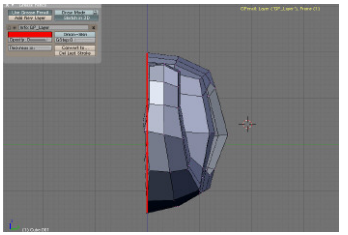
Step12. Add another Loop cut just inside the outer rim of vertices.

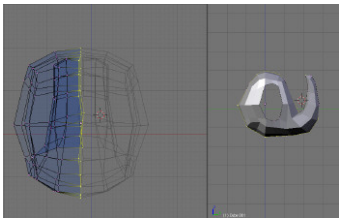


by Sandra Gilbert

by Sandra Gilbert

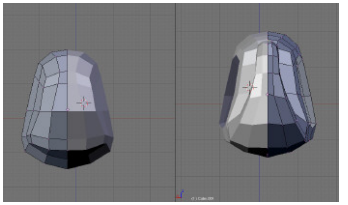
Step13. Now we get to do a bit of adjusting and eyeballing it from all angles. The Proportional edit tool can be rather useful to help manipulate the vertices into a softer, more rounded look. While you are adjusting make sure to start moving the tail into a slimmer profile.





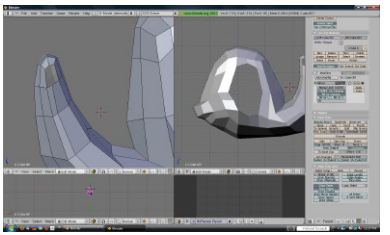
Step 14 & 15. Now is a good time to add your Mirror Modifier, so you can start seeing how it is looking overall. Continue to fine tune the shape.

When your model is sufficiently whale shaped, go to the tail section.

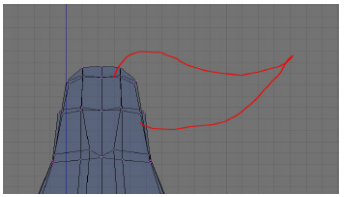


Step16. Delete the 2 faces and the tip of the tail.

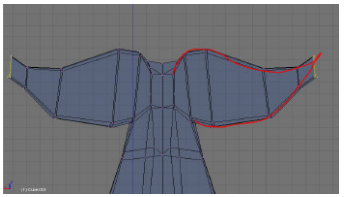
Note: It might be easier to work on the tail if you select all the vertices of the body and hide them (H)



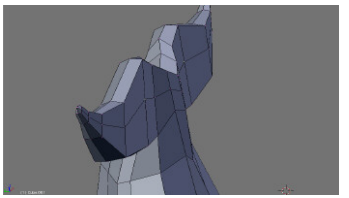
Step17. Using the Grease Pencil, sketch out an outline for the tail fluke.



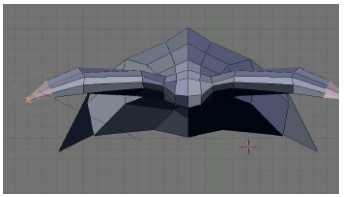
Step18. Now start extruding the vertices out to match the fluke outline.

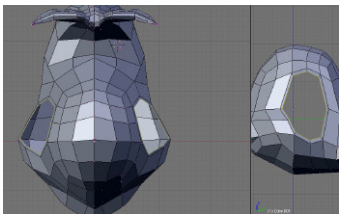


Step19. You'll notice that the fluke looks really flat and straight.

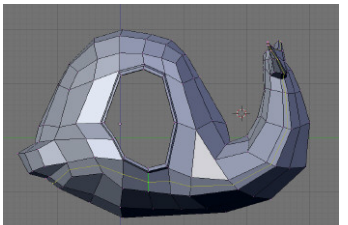


Step 20 &21. We'll use the Proportional Editing tool to create a more natural curved look.

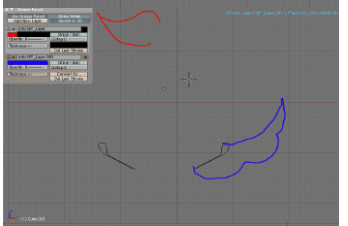




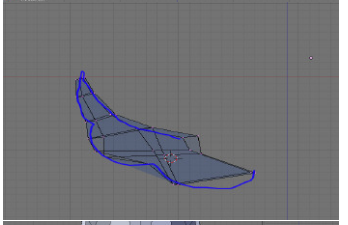
Step22. Using the same technique we used for the tail fluke, we will make some side fins. Add a Loop cut under the eye.



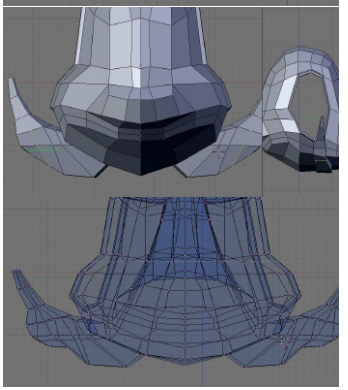
Step23. Remove a few faces below and slightly back from the eye and then draw your fin reference.



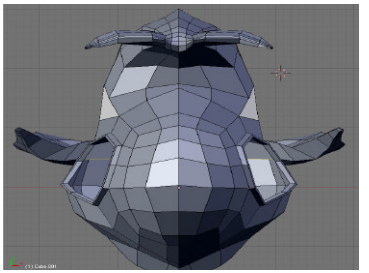
Step24. Extrude the fin out, following your outline.



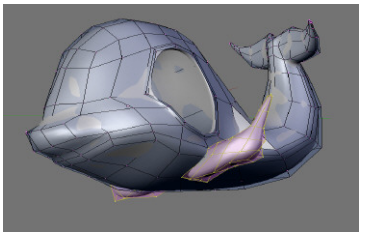
Step 25 &26. Depending on how you drew your fin and then extruded it, you might need a Loop cut or two to get it to look right.



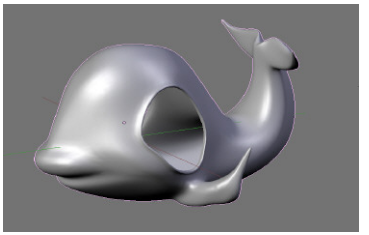
Step27. Use the Proportional Edit tool to rotate and curve the fins back towards the tail a little.



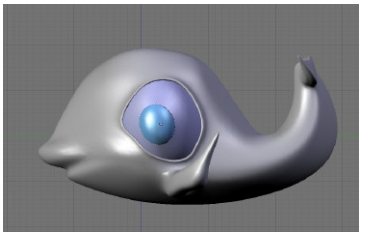
Step28. Okay, now you can go ahead and add a Subsurf Modifier. Look at your model from all angles, and smooth and adjust vertices as needed.



Step29. Depending on your sketch, your whale should look more or less like this.



Step30. At this point I added in an eye model that I keep around just for this reason.

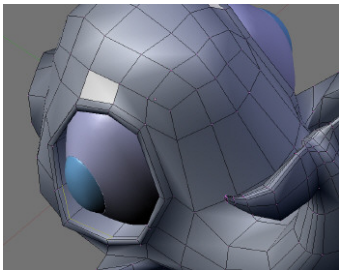


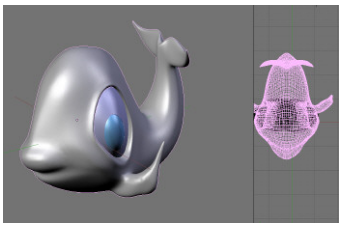
by Sandra Gilbert

Note: There are many Pixar style eye tutorials online, that you can use to create a small library of ready to use eyeballs for projects like this.

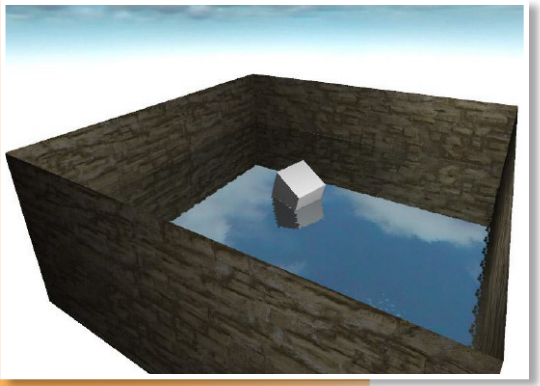
Step31. Now I don't know about you, but I always end up needing to adjust eye sockets once I place the eyeball. (31)

Step32. And here is my final little whale. I hope you had fun creating yours ■





by Sandra Gilbert



Realistic Water Environment

By SpewBoy

Introduction

This is my first proper tutorial, it aims to guide you through the steps of making a realistic water environment in Blender 3D. This water setup is for real-time use, great for games. In this tutorial I will show you, how to make the water reflective, how to add ripples, how to add underwater effects and much more. I hope that you will like it :).

What you will need

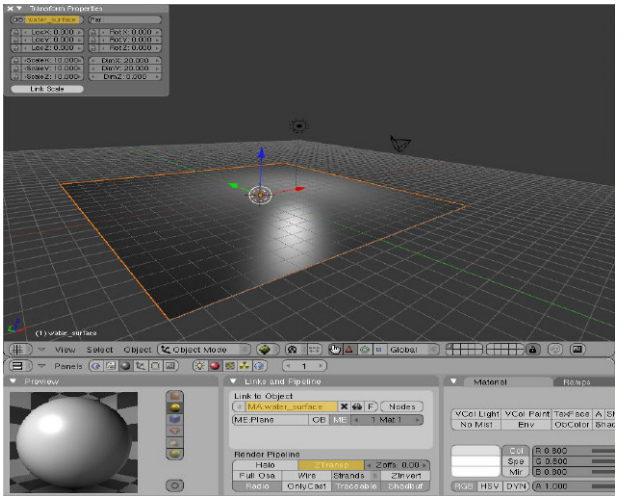
- Blender 2.49 or later.
- Some understanding of the game engine and general Blender knowledge. If you are confused by anything in this tutorial, you may need to acquire more Blender knowledge before following it.
- An animated water normal map (more on this later)

Let's begin!

Open up Blender and delete the default cube. Change the viewport shading type to "Texture" and in the Game Settings click "Blender GLSL Materials".

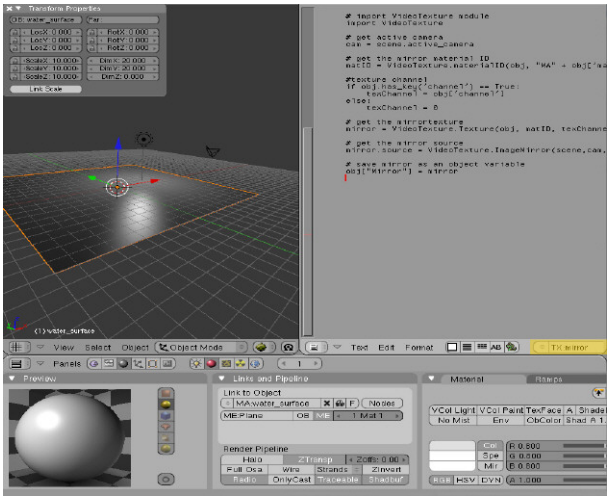
Press "Space" and select "Plane" to add a plane. Press "S" and then "Ten" to scale the plane up by ten times. This plane is our water surface, so press "N" and rename it to "water_surface".

Once this is done, add a material to the plane and name it "water_surface". In the material settings activate "ZTransp", to allow the plane to have transparency.



Okay. Now we will set up the logic and python for the water. In the logic section add an always and link it to a python controller. Add a property called "material", set it as a string and enter "water_surface". Split the 3d viewport in half by right clicking on the bottom edge and selecting "split area". Change the new viewport to a text one and click add new or select the script called

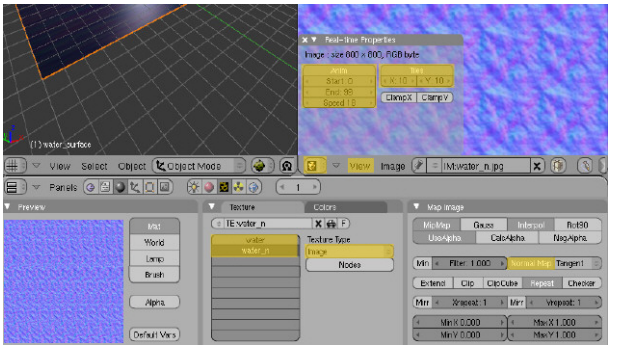
"Text". Now open the mirror.txt file included with this tutorial. Open it with notepad or something similar. Now copy the text and paste it into Blender's text window. Rename the script to "mirror".



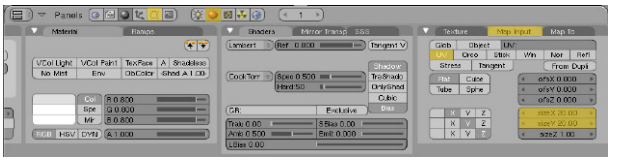
In the python controller we added earlier, type mirror as the python script. Go back to the always sensor and click on the button with the three dots. This means that the sensor will trigger every frame.

Go to the texture section of the water surface material and add a new texture. Set it as an image and click “load”. Now look on your disk for the texture included with this tutorial called “water.png”. Every reflective object in the game must have a unique “mirror” texture. The resolution or colour of the texture is not important, that’s why the one included is very small. I made it blue so it looks a bit like water in the viewport. The textures should have useful names like “water” and “water_n” for the texture explained bellow.

Now add the texture included called water_n.jpg. This is the normal map. It is what provides the ripple effects. In the texture’s settings activate the “Normal Map” setting. OR Look online for a tutorial on how to make an animated normal map texture yourself. There is a very good one here.



Now change the text viewport to a UV/Image editor viewport. Select the normal map texture and click “View” -> “Real-time properties”. Set up the real-time properties like you see in the picture bellow:



Now go to the “map to” tab for the water_n texture and deselect “Col” and select “Nor”. The “Nor” value underneath where it says “Mix” can be put up to 1.0 or down depending on how strong you want the ripples to be. Note that this is only for the ripple’s specularity. The actual distortion comes later and is done by using nodes.

Add a new texture to the water material. Browse for radial_alpha.jpg (included with this tutorial). Give it the name radial_a, and then delete it again.

by SpewBoy

by SpewBoy

This is just to get it registered as a proper texture. We will use it in the nodes for the water fresnel.

And speaking of nodes, I believe it's time for the node part of the tutorial. This isn't a section that I can describe in great detail, as there are so many steps to making it all work and if one section isn't done quite right, the whole thing won't work properly.

So, in your water's material settings click the nodes button. Once this is done a little red selection box will appear further down. Instead of adding a new node, click the arrows and select "water_surface". Now change the UV mapping window to a node window. This is the tricky part. You will see two "windows" or nodes so to speak. The one we want to keep is the one on the right. This should always be to the right. The other one can be deleted for now (select it and press delete). This next part I can't explain, so I have included a large picture of just how all the nodes should be set out (*node image is included in the blend file download*).

The picture is called "node_setup.jpg". You will need to look at this picture and try to copy every node and every setting within it. To make it easier I have put a note at the bottom of them to say what each node is, so you can find them easily when adding that type of node. You can look at the final blend as a reference too if you get stuck. To link two nodes drag from one circle to the other. To delete a link between two nodes, click and drag across the link. To add a new node click "Add" and select the node type you think is appropriate. The explanations in the image should help you find the necessary node type.

Once this is done, press "P" and give your water a test run. Hopefully you will see animated ripples in the water (don't panic if you're not, just continue), but what's this? The water appears completely blue! To fix this, press "TAB" while the water is selected and press "A" to select all faces. Change the node window to a UV window again and you will see a plane with four vertices. Press "A" again in the UV window to select all the vertices and press "M" to mirror the mapping on the X axis. Once this is done, you may need to rotate the mapping too so press "R" and type "180". This will rotate the mapping 180 degrees.

Now add a cube somewhere near the water and press "P". Hopefully now, the water will be reflecting the cube and you should be able to see ripples. If not, look at your node setup more closely, or if it's reflecting wrongly, look at you UV mapping setup and mirror it some more or rotate it etc. Note that the water is still mostly very blue. We will soon fix this.

I won't tell you how to make a sky dome or anything, or it will make this tutorial very long, but basically, the water is blue because it has nothing to reflect in the blue areas, so you need to make a skybox or sky dome and give it a sky texture. There is one in the final blend if you want to use that.

Now would probably be a good time to add a seabed. To do this just add a plane under the water and give it any old sand texture. One is included. Now when you play the game, you should be able to see the seabed underneath at close range, and when you play from further away the water appears opaque. This means that the fresnel texture and camera depth data is working.

Add four more planes around the water for walls. Once they are positioned around the water, and look like walls select them all and press "J" to join them all together. Then press "U" when all faces are selected and unwrap it using a cube setting or something similar. Add a new material and texture to the walls. A brick texture is provided.

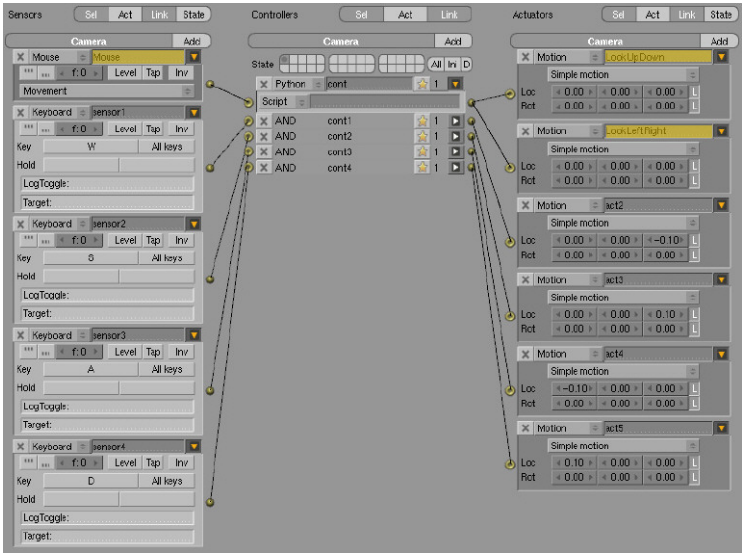
Now when you press “P” you should see the walls being reflected too. This is basically where the outside work stops and the underwater work begins.

Select the camera and press “Shift” + “S” and select “Curser -> selection”. This will snap the curser to the camera. Now press “Space” and add a cube. Scale the cube by around 0.1. Parent the cube to the camera (select cube, then camera and press “Ctrl” + “P”. Now create an empty in the same way, but instead of parenting it to the cube or camera, we are going to parent it to one of the cube’s vertices. To do this, select the empty, then the cube, press “TAB” and select a vertex and press “Ctrl” + “P”. We need to parent the empty to a vertex so that it follows the camera but does not rotate with the camera. Normal parenting affects rotation, but vertex parenting doesn’t for some reason. We need to keep the rotation even because the empty must always be facing the same way, as it is measuring where it is in relation to the water’s surface.

Now select the water and add a new property in the logic panel. Call the property “water”. Select the empty again and in the logic panel add a property called underwater and make it a bool. Now add a ray sensor that points in the –Z direction (downwards) and a sensor that points in the +Z direction (upwards). Make both the ranges as high as they will go and enable the x-ray option. In the property area, type “water”.

Add two controllers and two actuators. The first actuator should be a property actuator that changes the underwater property to true, and the second should change the property to false. Link them up accordingly. All this means that when the empty detects the water underneath it, it sets the underwater property to false. If it detects the water is above it, it will set the underwater property to true.

Now add two property sensors, two controllers and four actuators. The first two actuators will activate the underwater effects; the second two will deactivate the effects. So the first sensor should sense if the underwater property is true, the second should sense if it is false. Once the sensors are done, open up the file called “blur_radial.txt” and copy and paste the contents into Blender’s text window. Name the new script as blur_radial. Do the same with the file called “fog_underwater.txt”, except call it fog_underwater. In the first actuator, set it as a custom 2d filter and type blur_radial. For the second do the same with fog_underwater and make the pass, pass 1 instead of pass 0 (as 0 is taken by blur_radial). Now for the other two actuators, make them remove 2d filter actuators and set the first one to remove pass 0 and the second to remove pass 1.



by SpewBoy

Link them up so that the first two new actuators are linked to the third controller, which should be linked to the first property sensor. It should be clear how to link up the rest now. Oh, and make the cube invisible in the physics settings. Now we need to get the camera to move. Look at the screenshot bellow to see how the logic should be set up:

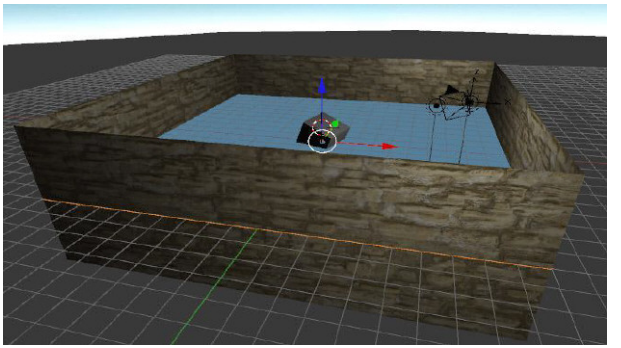
The highlighted names MUST be like they are in the picture. Now add another script in the text window, this time from the "mouselook.txt" file. Name it "mouselook". Now write "mouselook" in the empty script box you now have in the logic panel. Press numpad 0 to go into camera view and press "P". You should now be able to fly around using the W, A, S, D keys and use your mouse to look around.

If you fly underwater, everything should go foggy and the edges of the screen should be blurry and when you exit the water the effects should disappear. If not, something has gone wrong in the empty's logic. The last step is to add another plane just slightly under the water, facing downwards. Make a new material for it and give it the water_n texture, tiled 20 times. Apply the texture as "Nor" (not "Col"). Colour the texture blue-ish and play the game. When you look at the surface from underwater you should see ripples. If not, mess around with the map input settings some more.

That's all folks

And that pretty much concludes this tutorial. If you have any questions, email me at: etphonehomeplz@hotmail.com

Now all you need to do is make a game world and a functioning game and you're on your way to becoming a Blender pro like me xD (kidding, I'm not a pro. I've got a long way to go yet).



Thanks goes to martinsh for the 2d filters ■

-SpewBoy

by SpewBoy

Kaldewei

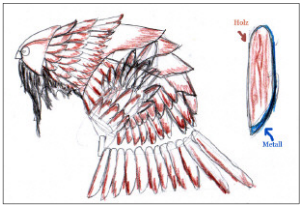
By Christoph Aka Dracio

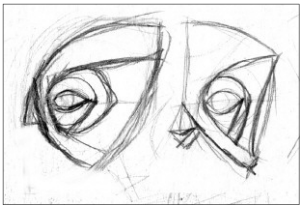
Introduction

How I get to that monster between a tank, some armor and an owl, I am really unable to tell you. I just tried to create designs that looked unique. Fairy tales, legends, myths and folklore around the world influence my work, but so do artists like Brian Froud (DarkCrystal), Alan Lee (Lord of the Rings), Tony DiTerlizzi (Spiderwick), Jim Henson (Muppets) and Mike Mignola (Hellboy).

I also develop characters with storytelling in mind. The story of Kaldewei is simple and you are able to see it on Kaldewei himself.

He was once a human doctor, an alchemist and a wizard. He shall get the death penalty because he's tortured people to death, but he was able to bind his soul to this armor.





I made the armor in an owl shape because the owl is a symbol of knowledge and a creature of the night at the same time. He also has no lower jaw. This is a little allusion to zombies and the fact that he is undead. For his lower jaw, he got the long wild beard that reminds that he has been a human many yeas ago.

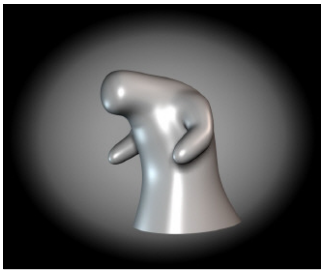
I always want to create monsters with personality, character and big piece of fantasy. Kaldewei is a very good example of when all those strange elements link very well together.

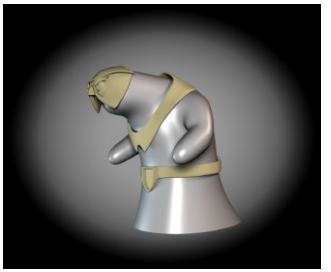
What's it all about?

I had made Kaldewei for a short movie I am going to make. I want to make a really good movie. For this project I wanted to use multiple characters and sets.

The Model

To translate the ideas of Kaldewei into a model, I created a very coarse model of Kaldewei's body. It didn't have any details; just the form of his proportions. Now I am able to build the armor.

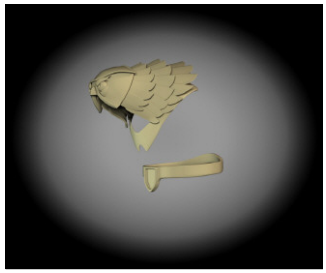


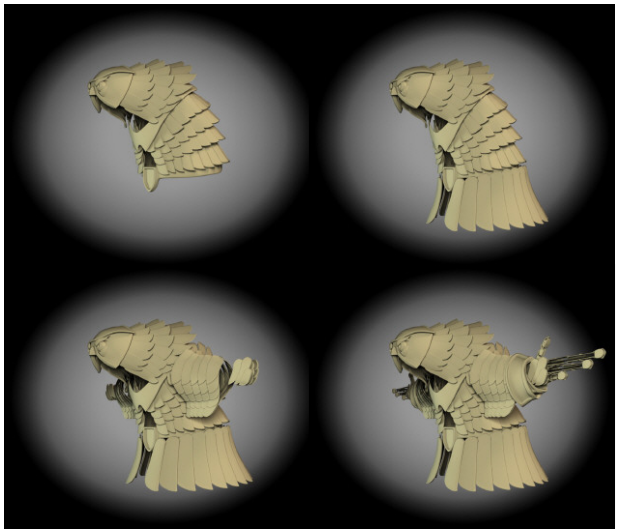


The mask is a major element of the whole set of armor and took a lot of time, but a good sketch helps a lot. After finishing the mask, I made the shoulder strap and the belt. They are not leading pieces in the finished model, but they are important pieces, because they stabilize the armor and split the plate-armor into 3 parts.

by Christoph Aka Dracio

What is missing are the plates. To pose and fit each plate on its own is very awkward and complicated to change at a later time. So I modeled one plate, prepared it for UV mapping and split the border from the rest by using two materials.

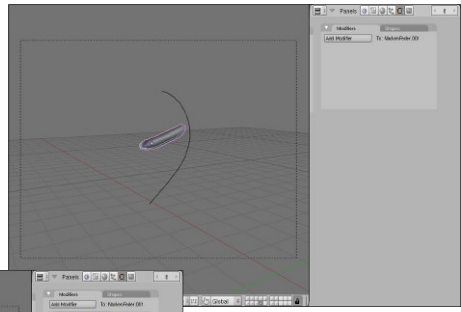


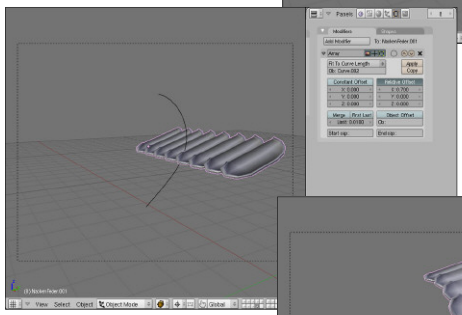


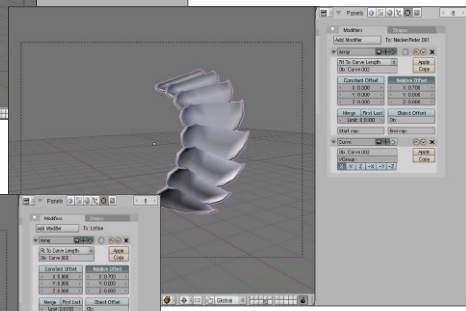
This plate is the base for all others in the model. I had made the plate-armor in strips. I used a plate, a “curve," the array modifier, the curve modifier and some lattices.

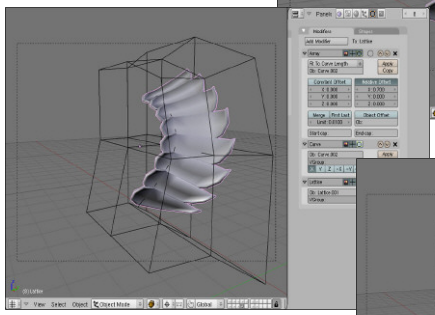
I placed the curve on the coarse model at the place where the strip has to be.

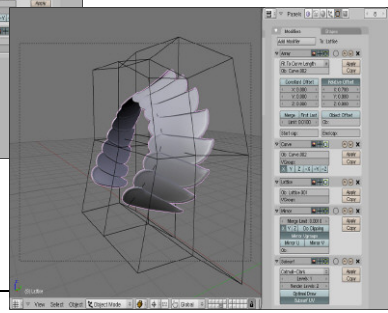
Array modifiers are able to make plates along the length of the curve. The curve modifier put our plates in the form of the curve. To finish the job we use a lattice to put our plate strip in the right form.











by Christoph Aka Dracio

Texture and the UV

The variety of details in the wood is from the textures. A texture has to fit three requirements. First, it has to have a huge resolution (2500 x 2500 pixels); second, a lot of details like knotholes, cracks and so on; and third, it shouldn't have perspective blur or compromising blur.

I set up the UVs in two layers. The first layer is made for the texture images like the wood or the rust. It is usual that unwrapped UV elements will overlap.

The second UV layer

The other UV layer is for pre calculations like "Normalbump" or “Ambient Occlusion,” and because of that the unwraped UV elements are not able to overlap.

Of course, I could use “Texture Bake ” or "Projection Painting “ and change my texture to fit in one UV layer but this has 3 drawbacks:

1 You get a lot of texts very fast

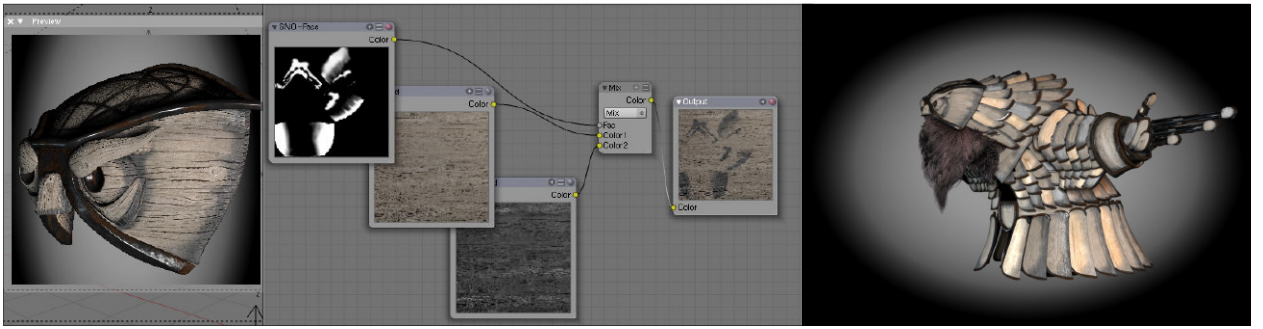
2 Many textures make mistakes

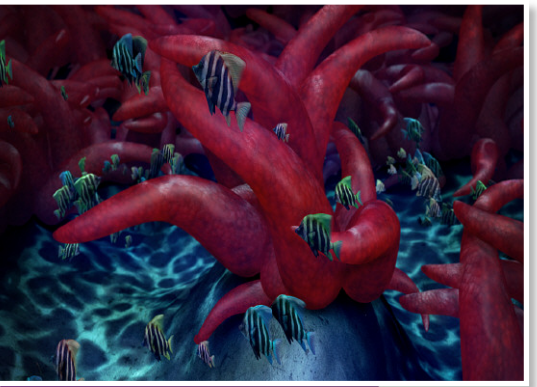
3 You always lose detail by changing a texture that much.

The Rig

I haven't started rigging the character. Because I have not finished the model completely and I need to make some more tests. My requirements for the rig are that each limb is able to extend and to shrink, and that the behaviour of the plates is physically correct. I want to make a sort of ragdoll system for the plates that lets them swing around.

You will find my project thread here with the latest news. And thank you for all your submissions and your good advice ■





Sea Anemone

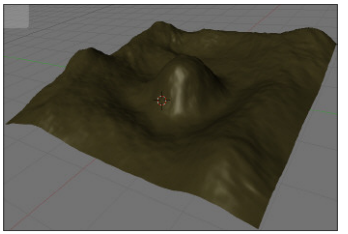
By Arland B. Woodbam III

Introduction

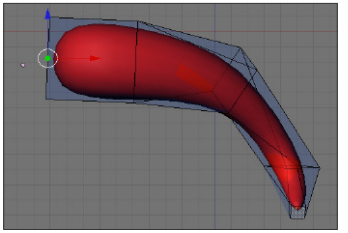
Thanks goes out to the guys at cgcookie.com for their video tutorial on “Painting” with Instanced Particles, from which this tutorial drew its inspiration.

Object instancing using particles can be a powerful feature in Blender that is often overlooked. While not as strong as something like Maya’s paint effects, you can still quickly accomplish things that would take many hours of tedious hand placement.

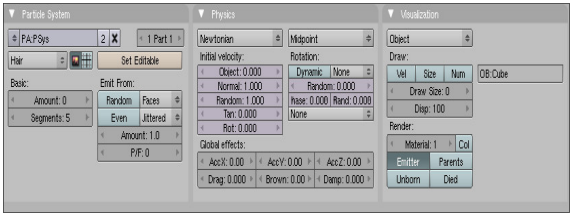
For this scene we are creating a sea anemone growing on the ocean floor. Lets start by creating a basic plane and subdividing or use multi-res. Then use Blender’s sculpt-mode to build up a very simple sea floor.



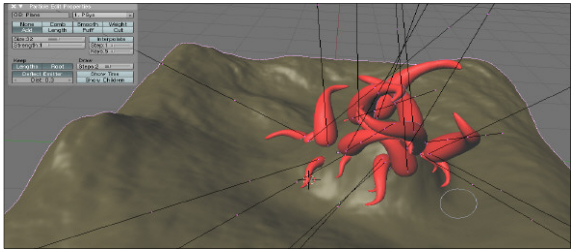
Now we need a stalk of the anemone to duplicate. This is a simple box that has been extruded and twisted a few times with sub-surf thrown on top. The important item to ensure here is that the origin point is located at the base of the stalk. This will ensure that the base of the “hair” particles we create will be at the base of the stalk. Don’t worry if your rotation is off. You can still edit the base object after using it in the particle system.



So let’s get to the fun part, painting the particles. First we need to apply a particle system to our sea floor. Use a hair type system with the amount set to 0 as well as a Normal and Random of 1.0. Also set the visualization mode to object and put the name of your particle in the OB field. I just left the default name of cube for mine.

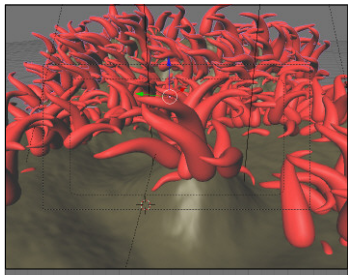


Now switch from object mode to particle mode and press the “N” key - this will bring up the tool box for particle editing. Select the Add button and set Strength to 1, then you can paint your particles onto the surface of your object. To delete particles simply switch to the Cut button.

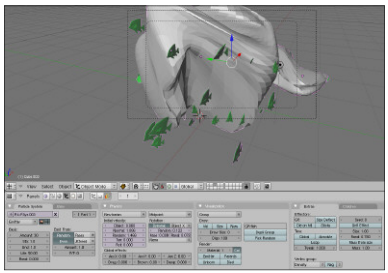


by Arland B. Woodbam III

Just have fun here and create what you want your image to look like. The final result here includes a second plane in the background to remove the open feeling of the camera shot.

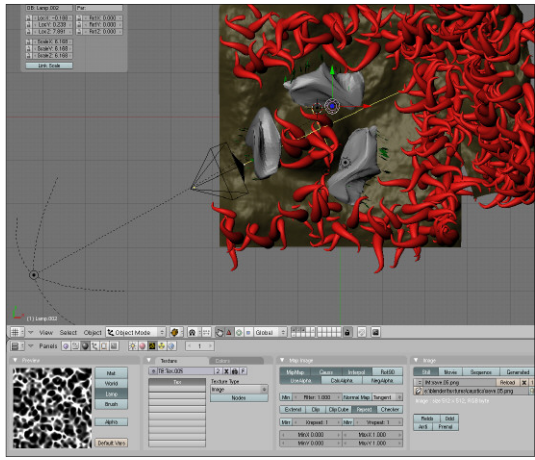


If you want more variety of shapes, then you can apply what we are going to do now with the fish. Here we have a relatively simple model of a fish which has been given the name of fish.



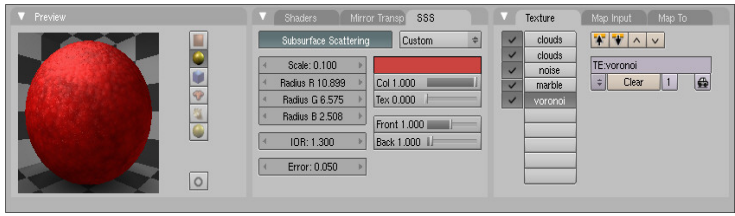
Now for the scene we want fish of different colors, so simply duplicate the fish and apply a different texture to each one naming the duplicates fish.001, fish.002, etc. This would also be a good time to modify each mesh slightly, although I have only changed the texture. For the emitter object we create an icosphere and use the sculpt mode to deform it.

This is to create the feel of a school of fish, so they clump together. The particle system is a very standard system. The only real difference is that the visualization is now set to group mode and the GR field has the name of the first object in the set "fish".

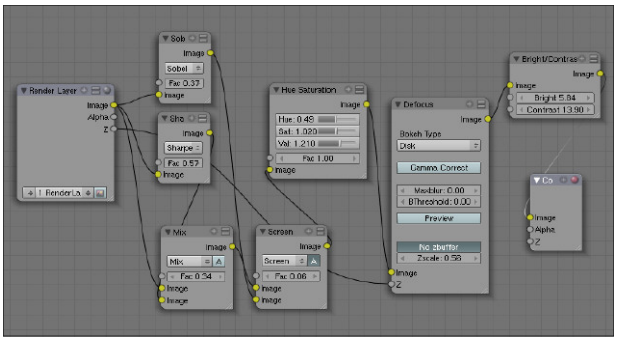


The lighting consists of a hemi light situated behind the camera which gives the underwater camera light effect. Basic ambient occlusion and mist to enhance the shadows with the AO set to a blue clouds, give the underwater feel. Then two spot lamps with a caustic texture map to create the feeling of light coming through the ocean surface. Finally a few low level lamps to lighten up a few areas.

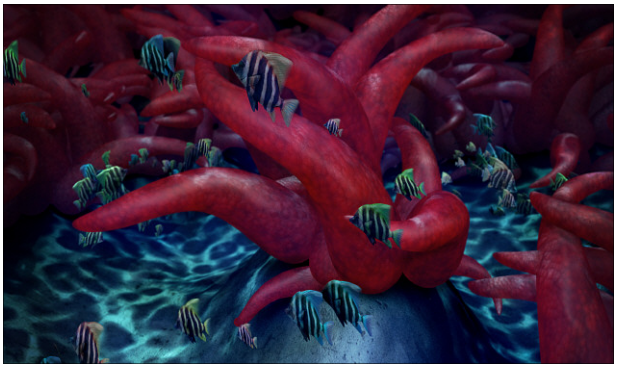
All of the textures are pretty much just a color map and a bump map with the exception of the stalks of the anemone. The stalks are many layers of procedural textures with subsurface scattering turned on. This part is very much a trial and error procedure until you get the look you want.



Finally the image is rendered out using nodes to enhance the edges, create a slight depth of field effect, and some tint adjustment.



The image was then rendered twice, once with the fish and once without. This was done so that the fish could be easily masked out in photoshop, because some will end up overlapping or looking too similar. The only other post-processing was to paint some light highlights and shadows on some of the stalks.



Hopefully this gives a bit of an insight into my process for creating this scene and motivates you to try out the particle instancing in Blender. I think it is an overlooked feature that allows you to create scenes that would take hours in much less time ■

Arland B. Woodham III aka Barry

I am a Graphic Specialist for a company that creates military training courseware.

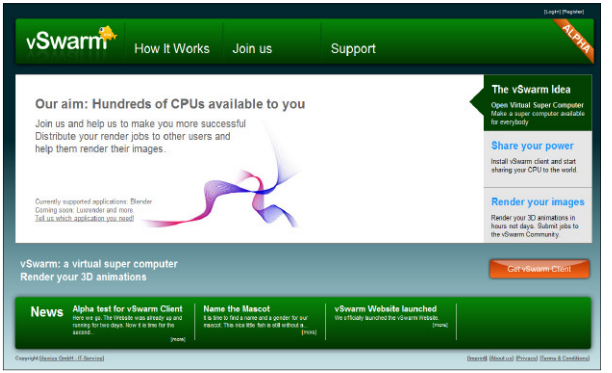
Website: http://www.cartoon-combat.com

Sea Anemone

by Thomas Schlüter

Introduction

Ever thought of how nice it would be to have a render farm for processing your animations? Or maybe you tried to set up your own small render farm? We did. But the farm was too small and too slow, upgrading it was too expensive and managing it was too time consuming, so we thought about a better solution and finally came up with a virtual render farm as a community project.



After some months of development we finally launched vSwarm - a community-based distributed rendering project. It's free and all of you are invited to participate.

Back to the beginning of the story: Why did we start vSwarm? To solve a problem many of you might have. The need to handle capacity peaks while rendering Blender animations. As we are two guys working a lot with Blender doing animations, it happens regularly that we need a lot of computing power all at once. But the rest of the time our CPUs are just twiddling thumbs (you know that, don't you).

Looking at the problem in detail, we realized that the root cause for it was our inability to handle capacity fluctuations.

vSwarm solves this problem as follows: As a member of the vSwarm community you can give spare capacity to the community (i. e. share our CPU power while it is idle) and ramp up our CPU capacity (i.e. use other client's CPUs) when you need to by submitting your jobs to the community.

Users sharing their CPU will receive priority points. The priority influences the number of CPUs available to a member when submitting a job to the vSwarm community.

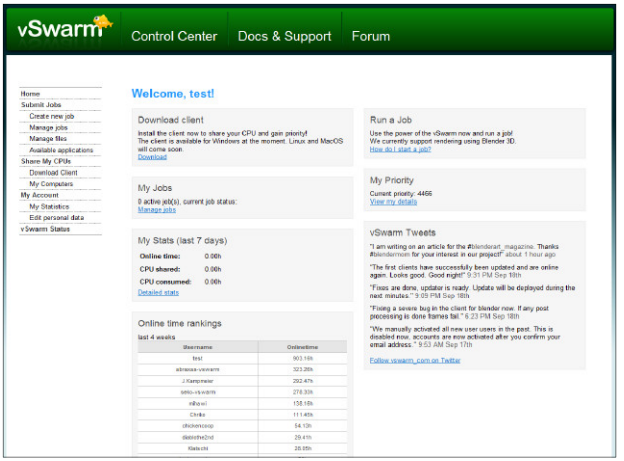
Right now, the project is still in an early stage, but growing and developing fast. We just included LuxRender into the client. And the next steps will be to build a Linux and a Mac version of the client.

One other goal for us in the near future is to generate some cash from the project ,as we are almost working full-time on the development of the project. To secure future development we have to find a way to cover the expenses of the project otherwise we cannot dedicate our resources almost full-time to it.

The first proposal for financing the project we would like to discuss with the community, is that companies, freelancers and everybody else who wants, can buy priority credits to get their jobs done faster. After paying for infrastructure and development, the rest of the money will be given to users running the client and to selected Open Source projects. But we won't stop the free part of the project.

by Thomas Schlüter

If you have any other ideas or comments about the financing of the project feel free to contact us. As this is a community project we invite you to influence the project by commenting on all aspects of the development of vSwarm.



That's it on why vSwarm was developed. Now, we want to thank everybody already active in the vSwarm community for their feedback and in the following we give you a first glimpse on how you can use vSwarm for your project.

How vSwarm works

vSwarm consists of two parts: a client and a web-based back-end. The client is installed on your computer and does the rendering. The web interface is used to submit jobs to the community.

The client in detail

The client is based on a Java GUI which controls the rendering. The rendering is done in a Linux environment running in VMware Player. Within this virtual machine we are running version 2.49a of Blender at the moment. The client also handles the up- and download of files to and from our servers.

vSwarm's Web Control Center

Via the web-based back-end the upload of the source files and job control is handled. In the back-end you also find statistics about your jobs and the participation of the community members.

Getting started with vSwarm

1. Register

Create your account at www.vswarm.com.

Now you have to decide whether you want to submit a job or download the client first.

2. Submit a job and get your blend file processed

After registering you can start using the back-end and you can submit jobs to the community.

Step-by-step instructions:

Make sure your animation meets the requirements

- Minimum RAM for clients is 256MB, so your animation should use a maximum of about 200MB. If the animation requires more RAM, you can start the job but will likely have failed frames and you will have to restart. We will change this soon.

by Thomas Schlüter

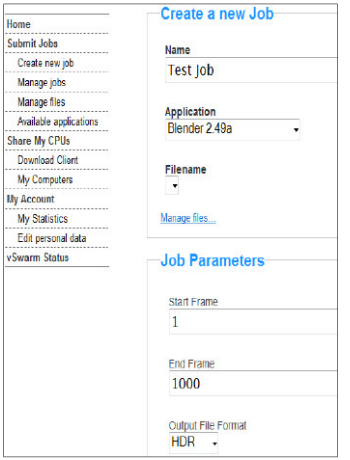
- Maximum processing time is 60 minutes for one frame. So make sure your frames finish within this time on a not too powerful computer. If one of your frames takes more than 60 minutes it will time out.

Set up your Blender file

- Select the camera for rendering as active camera.
- In the render panel set the resolution you would like to use for rendering.
- Set the options for your output format, e.g. JPEG compression or EXR bit depth.
- Set "xparts" and "yparts" in the render panel to about 5. This helps to utilize multi-core CPUs better if one or more parts take a longer time to render. And it allows the client to track the frame progress more exactly.
- Pack all your textures and other stuff into the file.
- Select "compress" from the file menu.
- Save the file to your HDD.

Upload your file to the vSwarm Server by using one of these methods:

- FTP: This is the best choice. It allows you to resume interrupted uploads and does not have a file size limit. You can find your login data for FTP at the bottom of the Manage files page.
- For small files it is possible to use the web upload. The size limit is 10MB per file and if anything goes wrong you have to start this again.
- After you have uploaded your file create the job in the vSwarm by submitting the following fields:



1 Job name: A name to recognize this job.

2 Application: Select the Blender version you would like to use.

3 Filename: Select the file you just uploaded.

4 Start frame: number of the first frame to render (e.g. 1)

5 End frame: last frame to render (e.g. 3000)

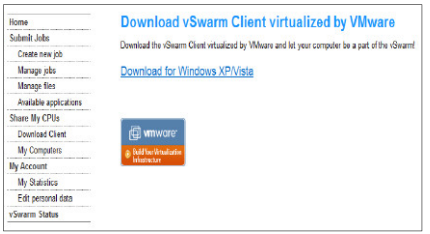
6 File Format: Select the output format you would like to get your results in. If the format has any options these are read from the blend file.

- Wait for your job to finish.
- If any frames have errors you can inspect the Blender output in the work units view and restart them.
- Download your result data.
- If you have rendered more than a few frames you must use FTP to download your files. Memorize the job id from the Web Control Center and login to the FTP server. Change to the directory "results" and then download all files in the directory of your job.
- Combine all frames to a movie with the application of your choice. I use Virtual Dub for this step.

by Thomas Schlüter

Download the client and share your CPU

If you want to help other members of the community, you can share your CPU and process their blend files. In order to do so, just download the vSwarm Client (around 400 MB as this is a full Linux-based virtual machine, the VMware Player and the Java GUI; it only supports Windows, we are working on Linux and Mac).



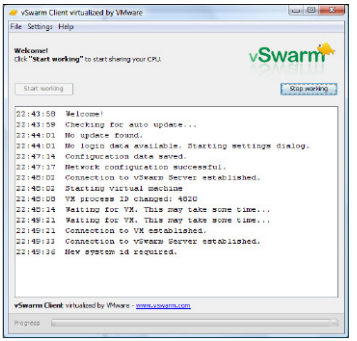
After the download is finished, start the installer and follow the instructions. Finally, reboot your system and then run the vSwarm Client. After you click the “Start working” button you will be asked for your user name and password. Now supply the credentials and you are part of the actively rendering vSwarm community.

Links:

www.vswarm.com the vSwarm Website.

Control.vswarm.com the Web Control Center.

Support.vswarm.com/wiki the on-line documentation ■



Now where's the "Under Water Lighting" button?

by Sandra Gilbert

At some point we have all wanted just such a button, but just like the fabled "Make my Render Beautiful" button, it's just not possible. There are way too many variables. But just because there isn't a pre-built button/option, doesn't mean you can't create some beautiful underwater scenes. Like most things in life, it just takes a little work.

First let's discuss water itself. You would think it would be as easy as just building the scene itself, water is after all clear. Yeah, you guessed it, its not that easy at all. The thing about water is that it is usually not clear at all. It often appears to have a slight color, and depending on the water in question, can appear murky due to sediment, plankton, algae and other microscopic debris. There may also be bubbles present depending on what else is floating about in the water. Luckily, most of this can be easily created in the material settings themselves. And any number of interesting bubble effects can be created with a creative combination of particles and soft bodies.

Note: A number of great material shaders can be found at:

1) Blender Open Material Repository

2) Blender 3d: Noob to Pro

So now we have realistic water, right?

Nope, not yet.

Now we have to tackle the whole lighting issue. Water has this annoying habit of reflecting, bouncing and diffusing light, producing of course, caustics and volumetric light. In a nutshell, volumetric light is produced by the light rays bouncing off all the microscopic debris floating in the water. The patterns of light (caustics) are created when that bounced, diffused light falls on objects in the water, such as rocks, fish or the sea floor, riverbed etc.

Now of course, we could export our scene to an external render engine, that handles volumetric light and caustics. The results would be as you expect and as a bonus, you would have plenty of time to go find something else to do while you waited for it to render. :P

But where is the fun is in that?

Instead, we are going to fake it. There is always a way to fake an effect. First we have to decide just what we are trying to fake. So what are we going for? Well, we need soft, fuzzy light beams and some pretty light patterns. It can't be that hard.

Now I know that fuzzy soft lights can be created with spotlights and halo options. But here I always run into problems. I don't know about you, but whenever I attempt to use halos, I end up with a huge fuzzy cone of light that completely blows out my image. Not a good underwater look, and we haven't even addressed the caustic part of the problem. This just might be harder than I thought.

I may not have a lot of luck with halos, but I am Queen of Google searches. There is a wealth of Blender information online, but oddly enough it took quite a while to find just what I needed. Eventually I found my solution in the form of an excellent, in depth tutorial at www.dnapixels.com, the tutorial not only covered underwater volumetric lighting but the underwater caustics as well.

Well there you go, step by step instructions on how to achieve that magical underwater lighting. Using a rather ingenious combination of special rotating discs and procedurals, the tutorial covers how to go about setting everything up, and as a bonus even covers the trickier subject of animating those light beams and patterns dancing on the sea floor ■

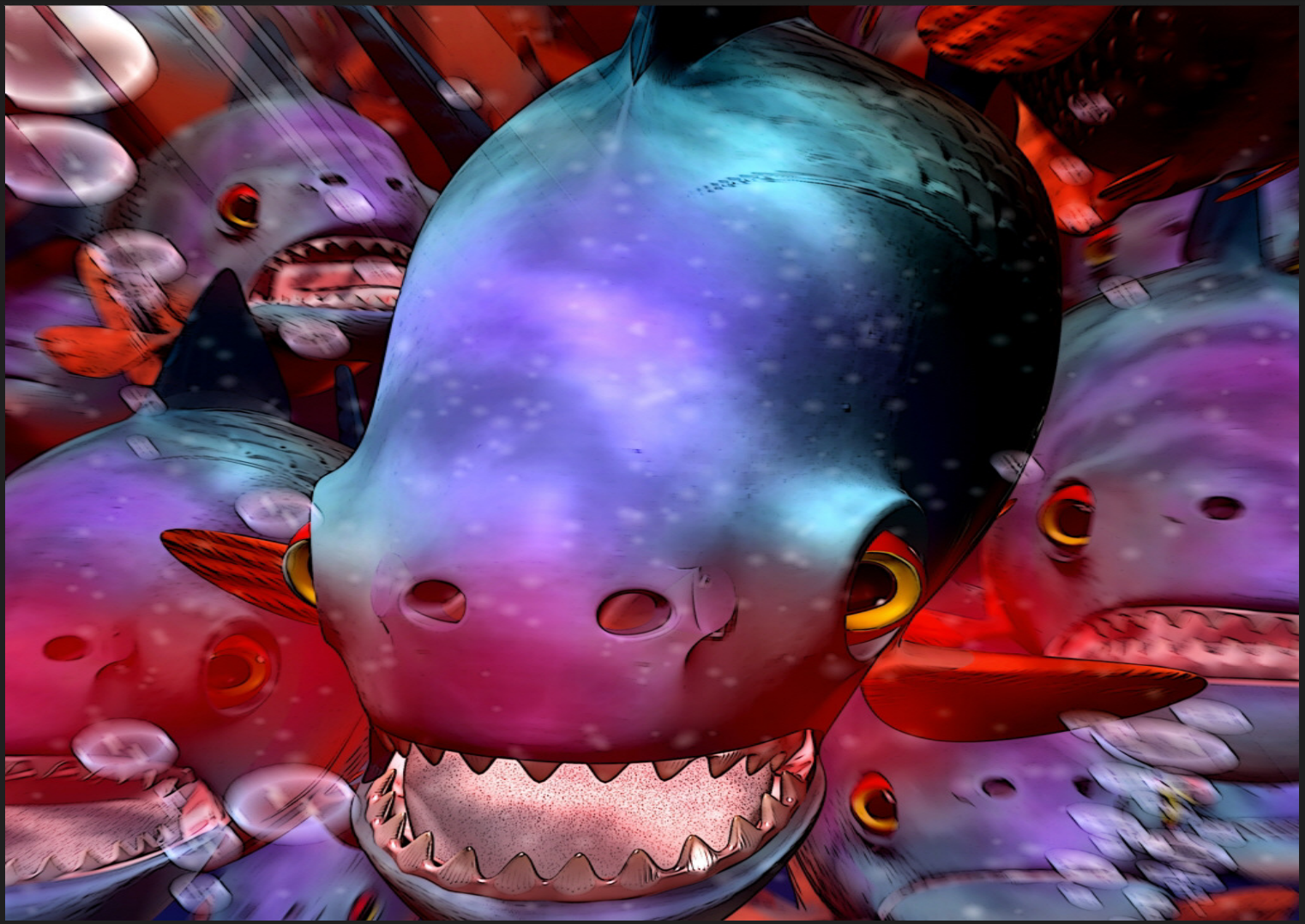
Rich & Dead - by Philippe Roubal

Hays - by Jeepster

Butterflyfish - by Davide Maimone

MonsterFish - by Jeremy Cox

Pirana - by Davide Maimone



Wreck - by JG Loquet

Helicoprion - by Osman Acasio

Ichthyosaur - by Alfie Urencio

Cooss - Lubomir Lopusek

Cracken - Tenchiken

Donkey Kong - by André Rubio

Want to write for BlenderArt Magazine?

Here is how!

1. We accept the following:

- Tutorials explaining new Blender features, 3dconcepts, techniques or articles based on current theme of the magazine.
- Reports on useful Blender events throughout the world.
- Cartoons related to blender world.

2. Send submissions to sandra@blenderart.org. Send us a notification on what you want to write and we can follow up from there. (Some guidelines you must follow)

- Images are preferred in PNG but good quality JPG can also do. Images should be separate from the text document.
- Make sure that screenshots are clear and readable and the renders should be at least 800px, but not more than 1600px at maximum.
- Sequential naming of images like, image 001.png... etc.
- Text should be in either ODT, DOC, TXT or HTML.
- Archive them using 7zip or RAR or less preferably zip.

3. Please include the following in your email:

- Name: This can be your full name or blenderartist avtar.
- Photograph: As PNG and maximum width of 256Px. (Only if submitting the article for the first time)
- About yourself: Max 25 words .
- Website: (optional)

Note: All the approved submissions can be placed in the final issue or subsequent issue if deemed fit. All submissions will be cropped/modified if necessary. For more details see the blenderart website.

Issue 25

"Winter Wonderland"

- Winter characters and scenes.
- Winter sports and activities (**scenes, images, animations, games*).
- Winter holidays (scenes, images, animations, games).

Disclaimer